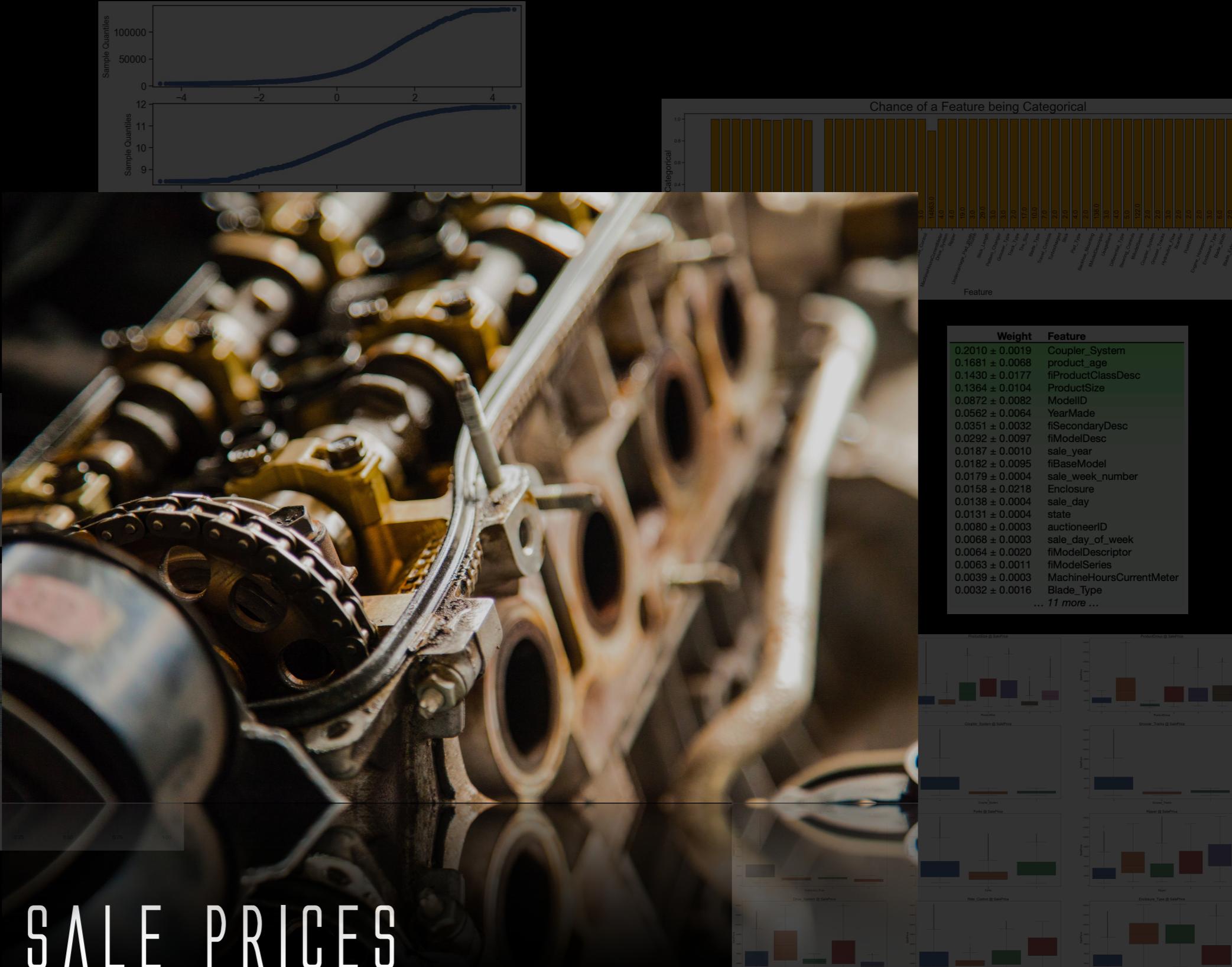
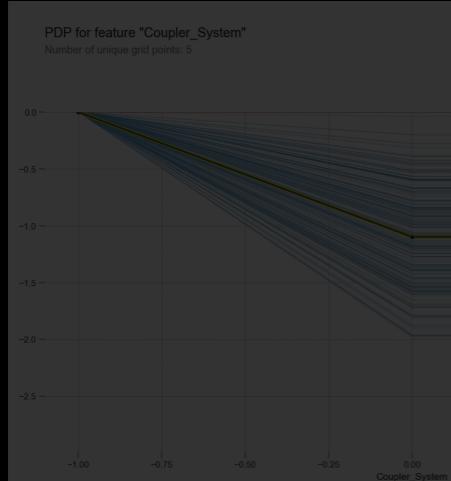


PREDICTING SALE PRICES

DATA PREP, FEATURE ENGINEERING, FEATURE SELECTION AND MODELLING PIPELINE





SALE PREDICTION

Objective

- **Prediction** : To predict Sales Prices at an auction.
- **Modelling Pipeline** : To build a Modelling Pipeline which could,
 1. Ingest the data.
 2. Clean and transform the data.
 3. Run Base Modelling (evaluate using k-fold cross validation).
 4. Fine tune the models hyper parameters.
 5. Evaluate the selected model on test data.
- **Model explainability** : A simple walkthrough to the various libraries which could explain the “Black Box” models results and interpretations.



SALE PREDICTION

Process

Imports

Imports

Importing the necessary libraries that would be required in this entire process. From data wrangling using *pandas* to data visualisations using *matplotlib*, *seaborn*. Various libraries for Algorithms such as *sklearn* and Model Interpretability *ELI5*.

1 Imports

```
In [1]: # General
1 import pandas as pd
2 import numpy as np
3 import os, ast
4 pd.set_option('display.max_colwidth', -1)
5 from tqdm import tqdm_notebook
6 import warnings
7 warnings.filterwarnings('ignore')
8 import pickle

# Visualisation
9 import seaborn as sns
10 from IPython.display import display_html
11 import matplotlib.pyplot as plt
12 from statsmodels.graphics.gofplots import qqplot
13 from IPython.core.display import display, HTML

#Algorithms
14 import xgboost as xgb
15 from sklearn.tree import DecisionTreeRegressor
16 from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
17 from catboost import CatBoostRegressor
18 from lightgbm import LGBMRegressor
19 from sklearn.neighbors import KNeighborsRegressor
20 from sklearn.linear_model import Lasso
21 from sklearn.model_selection import RandomizedSearchCV, KFold, GridSearchCV
22 import pingouin as pg
23 from sklearn.metrics import r2_score

# Model Explainability
24 import shap
25 import eli5
26 import lime
27 from pdpbox import pdp, get_dataset, info_plots
28 from IPython.display import Image
29 from sklearn import tree
30 import pydotplus

executed in 3.60s, finished 23:00:07 2019-12-23
```



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Path / Variable Initialisation

Initialising path,

- Where raw data sits
- Where prepared data will be stored
- Where Notebook Results will be saved

Initialising ***Target Variable***

Initialising ***Plotting Configurations***

2 Paths & Variable Initializations

```
1  ##### Path #####
2  root_path = '../'
3  data_path = root_path+'Data/'
4  raw_data_path = data_path+'Raw Data/'

5  complete_data_path = raw_data_path+'Data.csv'

6  prepared_data_path = data_path+'Prepared Data/'

7

8  notebook_results_path = root_path+'Notebook Results/'
9  os.makedirs(notebook_results_path, exist_ok=True)

10 #######VARIABLE INITIALISATIONS#####
11 TARGET_VARIABLE = 'SalePrice'

12

13 plt.rcParams['figure.dpi']      = 140
14 plt.rcParams['figure.figsize']  = (25, 5)
```



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Reading, Preparing &
Splitting Data

Reading, Preparing and Splitting Data

Reading the raw data and analysing the individual columns, to gauge further into what sort of treatment would that column require. Once done with analysing columns individually some analysis of the NULL or Unique counts present in the dataset.

Preparing the data builds up on the preliminary analysis and the understanding of the data thus far, which could mean parsing few columns into date-time, dropping few irrelevant columns and converting categorical columns to their label encodings or one hot encoding.

4 Reading, Preparing and Splitting the data

- ▶ **4.1 Reading the data**
- ▶ **4.2 Analysing the individual columns** ↗
- ▶ **4.3 Some Analysis**
- ▶ **4.4 Preparing the data**
- ▶ **4.5 Splitting the data**

[...]

[...]

[...]

[...]

[...]



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Reading, Preparing &
Splitting Data

Feature Engineering

Feature Engineering

In Machine learning, Data prep constitutes around 60% of the efforts and about 20% of the efforts are utilised in thinking about the features that could be engineered from the data which is available to us. The remainder for Modelling!

Sometimes engineering new features which are not just explainable to the problem statement but are able to extract useful information can improve accuracy.

The categorical columns doesn't usually contain data in numerical form, the only form comprehensible to the machine, for that reason alone those categorical columns need to be converted into the either their label coding or One Hot Encodings.

5 Feature Engineering

```
In [ ]: 1 train_data_EDA = train_data.copy()  
executed in 7ms, finished 16:58:54 2019-12-23
```

5.1 Adding Calendar Variables

```
In [ ]: 1 train_data_EDA['sale_year'] = train_data_EDA['saledate'].apply(lambda x : x.year)  
2 train_data_EDA['sale_month'] = train_data_EDA['saledate'].apply(lambda x : x.month)  
3 train_data_EDA['sale_day'] = train_data_EDA['saledate'].apply(lambda x : x.day)  
4 train_data_EDA['sale_day_of_week'] = train_data_EDA['saledate'].apply(lambda x : x.dayofweek)  
5 train_data_EDA['sale_week_number'] = train_data_EDA['saledate'].apply(lambda x : x.week)  
6 train_data_EDA['product_age'] = train_data_EDA.apply(lambda x: x.sale_year - x.YearMade if x.YearMade != -999 el  
7  
8 # Filtering out those DataPoints in which saledate was earlier than YearMade - Doesn't Make Sense, Right!  
9 train_data_EDA = train_data_EDA[train_data_EDA['product_age'] >= 0]  
executed in 21.8s, finished 16:59:17 2019-12-23
```



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Reading, Preparing &
Splitting Data

Feature Engineering

Feature Selection

Feature Selection

From a plethora of features available to the modeller (i.e You), selecting which features to feed into the pipeline is a craftsmanship that only experience can teach, but few basic techniques are already there which could be leveraged for selecting features.

These technique can vary for the various types of the data, i.e to select a continuous feature w.r.t a continuous target variable correlation might be a good evaluator on the other hand Annova η^2 can be used to measure the association of a categorical column to that of an continuous target.

▼ 5 Feature Selection

- ▶ **5.1 Feature Importance**
- ▶ **5.2 Correlation - Continous Variables**
- ▶ **5.3 Correlation - Discreet Variables**
- ▶ **5.4 Lasso Regression**

[...]

[...]

[...]

[...]



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Reading, Preparing &
Splitting Data

Feature Engineering

Feature Selection

EDA

EDA

Exploratory Data Analysis, is one of those crucial steps where most of the intuition about modelling and new feature engineering / transformation required in the data available can be charted out.

There is no hard and fast rule which says that exploring and making intuitions should be done at this stage alone, most of the times I usually end up making my intuitions at the Data Prep & Feature Engineering Stage. Sometimes I have to back and forth between these steps.

6 EDA

```
In [ ]: 1 train_data_SF = train_data_EDA[selectedFeatures].copy()  
executed in 20ms, finished 19:19:35 2019-12-17
```

6.1 Target Variable

[...]



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Reading, Preparing &
Splitting Data

Feature Engineering

Feature Selection

EDA

Modelling Pipeline

Modelling Pipeline

The concept of a Modelling Pipeline is as simple as, getting everything at one place in a structured fashion such that the pipeline functions can be reused to build models again and again when there is a

- Change in the Input (Data changes or selected features change)
- New model to be evaluated (Adding a new model into the pipeline or a new layer of modelling couch as Ensemble of Ensembles)

7 Modelling Pipeline

This class contains the entire pipeline for building base models:

Data Ingestion -> Data Cleaning/Transformation -> Feature Engineering -> Modelling

- ▶ **7.1 Read Data**
- ▶ **7.2 Modelling Pipeline**
- ▶ **7.3 Fine Tuning - HyperParameters**
- ▶ **7.4 Predicting the Test Set**

[...]

[...]

[...]

[...]



SALE PREDICTION

Process

Imports

Path / Variable
Initialisation

[Config]

Reading, Preparing &
Splitting Data

Feature Engineering

Feature Selection

EDA

Modelling Pipeline

Model Interpretability

Model Interpretability

Until unless Simple Regression(Linear, Polynomial, GAM etc) models are used which can in their inherent nature are quite explainable, Explainability of a model becomes a huge blocker in convincing business to actually use the model in production.

Here I discuss few basic packages which can be used for unraveling few black box models such as tree based (Boosting Trees, XGBoost, CatBoost and even NN).

- 1) ELI5 ("Explain Like I'm 5" - The closest thing I could find)
- 2) SHAP (Shapely Additive Explanations)
- 3) LIME (Local Interpretable Model Agnostic Explanation)
- 4) PDP (Partial Dependence Plots)
- 5) ICE (Individual Conditional Explanations)
- 6) Tree Interpreter (If tree methods are being used)



SALE PREDICTION

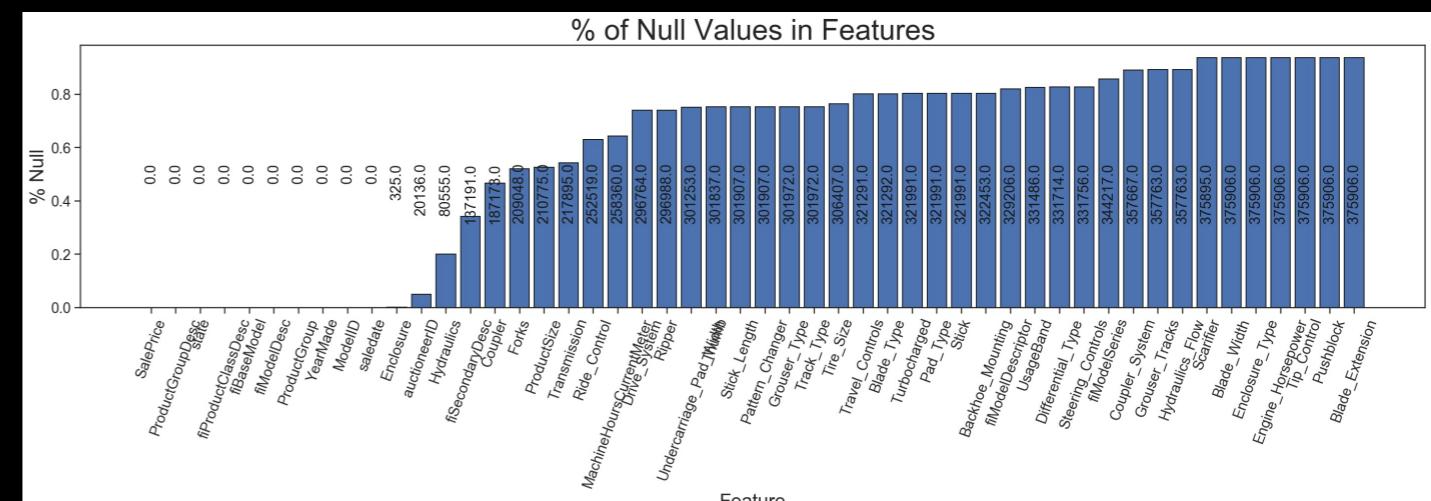
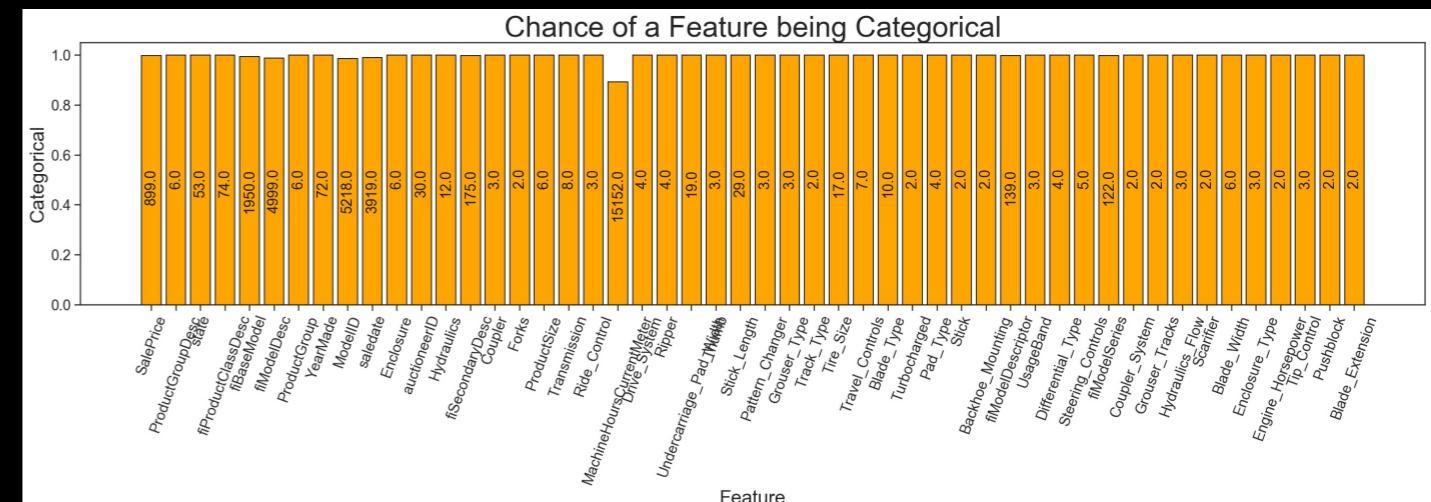
Understanding the data

Variable	Description
SaleID	unique identifier of a particular sale of a machine at auction
MachineID	identifier for a particular machine; machines may have multiple sales
ModelID	identifier for a unique machine model (i.e. fiModelDesc)
datasource	source of the sale record; some sources are more diligent about reporting attributes of the machine than others.
auctioneerID	identifier of a particular auctioneer, i.e. company that sold the machine at auction. Not the same as datasource.
YearMade	year of manufacturer of the Machine
MachineHoursCurrentMeter	current usage of the machine in hours at time of sale (saledate); null or 0 means no hours have been reported for that sale
UsageBand	value (low, medium, high) calculated comparing this particular Machine-Sale hours to average usage for the fiBaseModel; e.g. 'Low' means this machine has less hours given it's lifespan relative to average of fiBaseModel.
Saledate	time of sale
Saleprice	cost of sale in USD
fiModelDesc	Description of a unique machine model (see ModelID); concatenation of fiBaseModel & fiSecondaryDesc & fiModelSeries & fiModelDescriptor
fiBaseModel	disaggregation of fiModelDesc
fiSecondaryDesc	disaggregation of fiModelDesc
fiModelSeries	disaggregation of fiModelDesc
fiModelDescriptor	disaggregation of fiModelDesc
ProductSize	Don't know what this is
ProductClassDesc	description of 2nd level hierarchical grouping (below ProductGroup) of fiModelDesc
State	US State in which sale occurred
ProductGroup	identifier for top-level hierarchical grouping of fiModelDesc
ProductGroupDesc	description of top-level hierarchical grouping of fiModelDesc
Drive_System	machine configuration; typically describes whether 2 or 4 wheel drive
Enclosure	machine configuration - does machine have an enclosed cab or not
Forks	machine configuration - attachment used for lifting
Pad_Type	machine configuration - type of treads a crawler machine uses
Ride_Control	machine configuration - optional feature on loaders to make the ride smoother
Stick	machine configuration - type of control
Transmission	machine configuration - describes type of transmission; typically automatic or manual
Turbocharged	machine configuration - engine naturally aspirated or turbocharged
Blade_Extension	machine configuration - extension of standard blade
Blade_Width	machine configuration - width of blade
Enclosure_Type	machine configuration - does machine have an enclosed cab or not
Engine_Horsepower	machine configuration - engine horsepower rating
Hydraulics	machine configuration - type of hydraulics
Pushblock	machine configuration - option
Ripper	machine configuration - implement attached to machine to till soil
Scarfier	machine configuration - implement attached to machine to condition soil
Tip_control	machine configuration - type of blade control
Tire_Size	machine configuration - size of primary tires
Coupler	machine configuration - type of implement interface
Coupler_System	machine configuration - type of implement interface
Grouser_Tracks	machine configuration - describes ground contact interface
Hydraulics_Flow	machine configuration - normal or high flow hydraulic system
Track_Type	machine configuration - type of treads a crawler machine uses
Undercarriage_Pad_Width	machine configuration - width of crawler treads
Stick_Length	machine configuration - length of machine digging implement
Thumb	machine configuration - attachment used for grabbing
Pattern_Changer	machine configuration - can adjust the operator control configuration to suit the user
Grouser_Type	machine configuration - type of treads a crawler machine uses
Backhoe_Mounting	machine configuration - optional interface used to add a backhoe attachment
Blade_Type	machine configuration - describes type of blade
Travel_Controls	machine configuration - describes operator control configuration
Differential_Type	machine configuration - differential type, typically locking or standard
Steering_Controls	machine configuration - describes operator control configuration

The data taken for this project is from Kaggle competition for “Blue Book for Bulldozers”.

The complete data that I considered for this problem statement was of : **401125 rows × 52 columns**

← Target Column

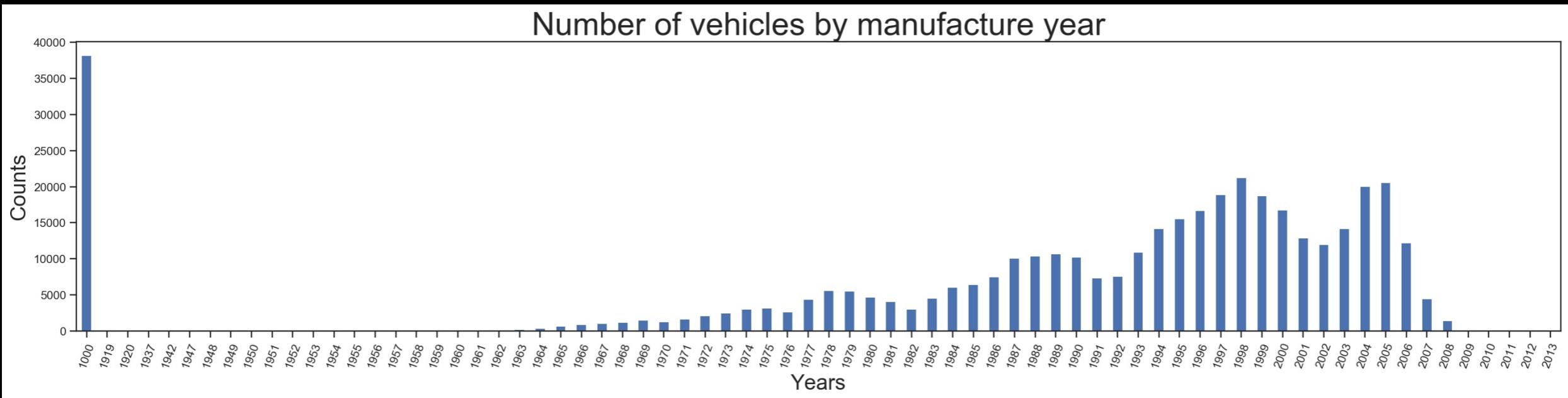




SALE PREDICTION

Understanding the data

Number of vehicles by manufacture year



```
In [176]: 1 ## Analysing the data - You gotta go through each column one by one
2 idx = 0
executed in 3ms, finished 16:56:29 2019-12-23
```

```
In [177]: 1 colAnalysed = complete_data.columns[idx]
2 print('Column Being Analysed : ', colAnalysed)
3
4 helperFunctionsHandler.display_side_by_side([complete_data[[colAnalysed]].sample(7),
5 complete_data[[colAnalysed]].describe(),
6 captions = ['Raw Data', 'Descriptive Statistics'])
7 idx += 1
8
executed in 57ms, finished 16:56:29 2019-12-23
```

Column Being Analysed : SalePrice

Descriptive	
Raw Data	Statistics
SalePrice	SalePrice
SalesID	
1735746	count 401125
	mean 31099.7
1796848	std 23036.9
2413133	min 4750
1640097	25% 14500
1839870	50% 24000
1589059	75% 40000
1468915	max 142000

- Target Column might be left skewed
- MachineID is not an unique identifier, as many vehicles were again auctioned
- ModelID is not an unique identifier, might be categorical
- datasource is an useless column, should be dropped
- auctioneerID is an categorical column
- There are 38185 data points in which YearMade is 1000 -- Junk Data
- Most of the data in 'MachineHoursCurrentMeter' is NaN -- replace it with -999
- UsageBand is categorical and should be ordered in order = ['High', 'Medium', 'Low']
- saledate is a date column. Parse it
- fiModelDesc, fiBaseModel, fiSecondaryDesc, fiModelSeries, fiModelDescriptor, fiProductClassDesc - All are categorical columns, and the missing values should be replaced with -999
- ProductSize is an categorical Size column
- state is an categorical column
- ProductGroup, ProductGroupDesc, Drive_System, Enclosure, Forks, Pad_Type, Ride_Control, Stick, Transmission, Turbocharged, Blade_Extension, Blade_Width, Enclosure_Type, Engine_Horsepower, Hydraulics, Pushblock, Ripper, Scarifier, Tip_Control, Tire_Size, Coupler, Coupler_System, Grouser_Tracks, Hydraulics_Flow, Track_Type, Undercarriage_Pad_Width, Stick_Length, Thumb, Pattern_Changer, Grouser_Type, Backhoe_Mounting, Blade_Type, Travel_Controls, Differential_Type, Steering_Controls, are categorical

Understanding the data is something which is very problem statement centric, it depends on the problem statement what exactly you want to analyse.

Analysing the Null and Unique values in a data frame can some time give good intuitions about the data and how the modelling might look like.



SALE PREDICTION

Feature Engineering

5 Feature Engineering

```
In [ ]: 1 train_data_EDA = train_data.copy()  
executed in 7ms, finished 16:58:54 2019-12-23
```

5.1 Adding Calendar Variables

```
In [ ]: 1 train_data_EDA['sale_year'] = train_data_EDA['saledate'].apply(lambda x : x.year)  
2 train_data_EDA['sale_month'] = train_data_EDA['saledate'].apply(lambda x : x.month)  
3 train_data_EDA['sale_day'] = train_data_EDA['saledate'].apply(lambda x : x.day)  
4 train_data_EDA['sale_day_of_week'] = train_data_EDA['saledate'].apply(lambda x : x.dayofweek)  
5 train_data_EDA['sale_week_number'] = train_data_EDA['saledate'].apply(lambda x : x.week)  
6 train_data_EDA['product_age'] = train_data_EDA.apply(lambda x: x.sale_year - x.YearMade if x.YearMade != -999 else  
7  
8 # Filtering out those DataPoints in which saledate was earlier than YearMade - Doesn't Make Sense, Right!  
9 train_data_EDA = train_data_EDA[train_data_EDA['product_age'] >= 0]
```

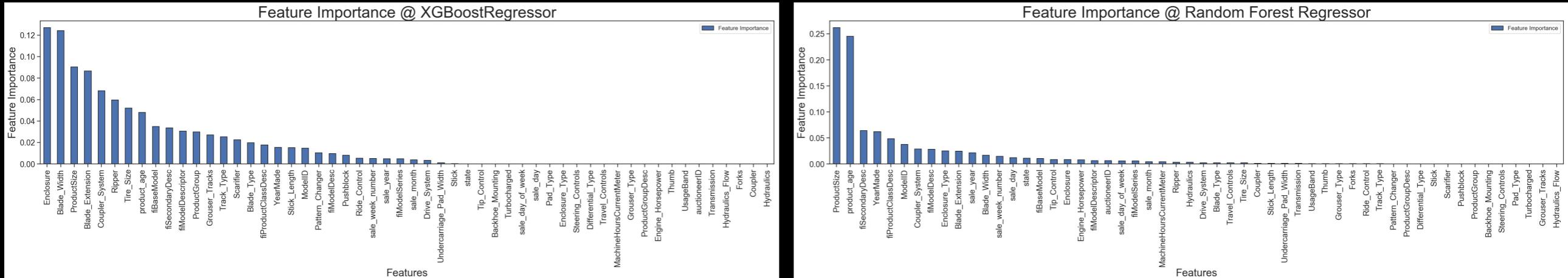
executed in 21.8s, finished 16:59:17 2019-12-23

1. Preparing the data can involve removing those columns which have absolutely no relevance the problem at, for ex ‘datasource’.
2. Now coming to feature engineering, I have dedicated a rather small chunk of my effort on the same, but quite honestly this is that aspect of Machine Learning which is probably where most of the creativity is involved and the “*Narnian Gateway*” for achieving higher score/accuracy for the model. Keeping that in mind I think this aspect deserves a separate post of itself. Although the features that I did engineer here were :
 - Calendar Variables
 - The age of the product - How old the product was at the time of auction from the year of manufacturing.



SALE PREDICTION

Feature Selection



Feature Importance

When there is a plethora of variables, it's usually a good call to remove few of those features which are not that relevant to the problem statement at hand. But how to figure those out?. In this project I have used following methodologies to select features, there are plenty more out there...

- Feature Importance - Just train a RandomForest or an XGBoost on the available data and see the feature importance of the variables



SALE PREDICTION

Feature Selection

	SalePrice
YearMade	0.220085
sale_year	0.039649
MachineHoursCurrentMeter	0.011550
sale_day	0.000552
ModelID	-0.034558
sale_week_number	-0.037413
sale_month	-0.037983
auctioneerID	-0.043810
sale_day_of_week	-0.052251
product_age	-0.250279

Correlation

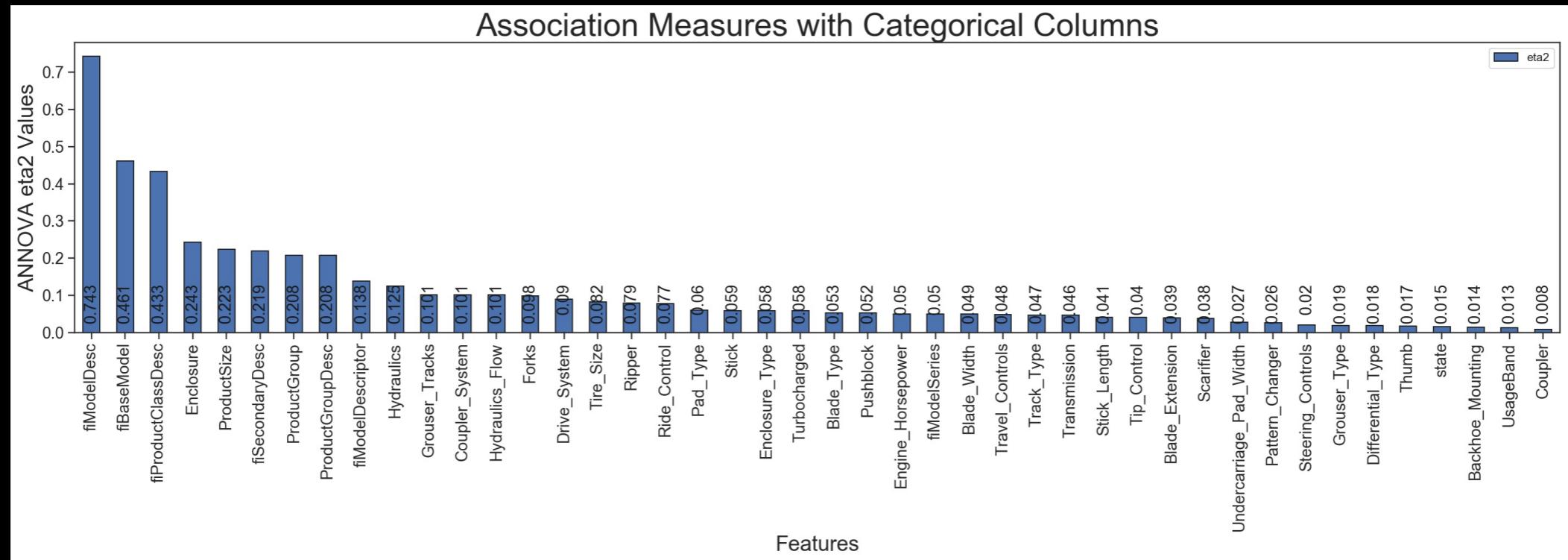
When there is a plethora of variables, it's usually a good call to remove few of those features which are not that relevant to the problem statement at hand. But how to figure those out?. In this project I have used following methodologies to select features, there are plenty more out there...

- Correlation [continuous-continuous] - To calculate how much of the variance of a Target Variable is being explained by the feature, look at the Pearson Correlation of the same.



SALE PREDICTION

Feature Selection



ANOVA η^2 Association Value

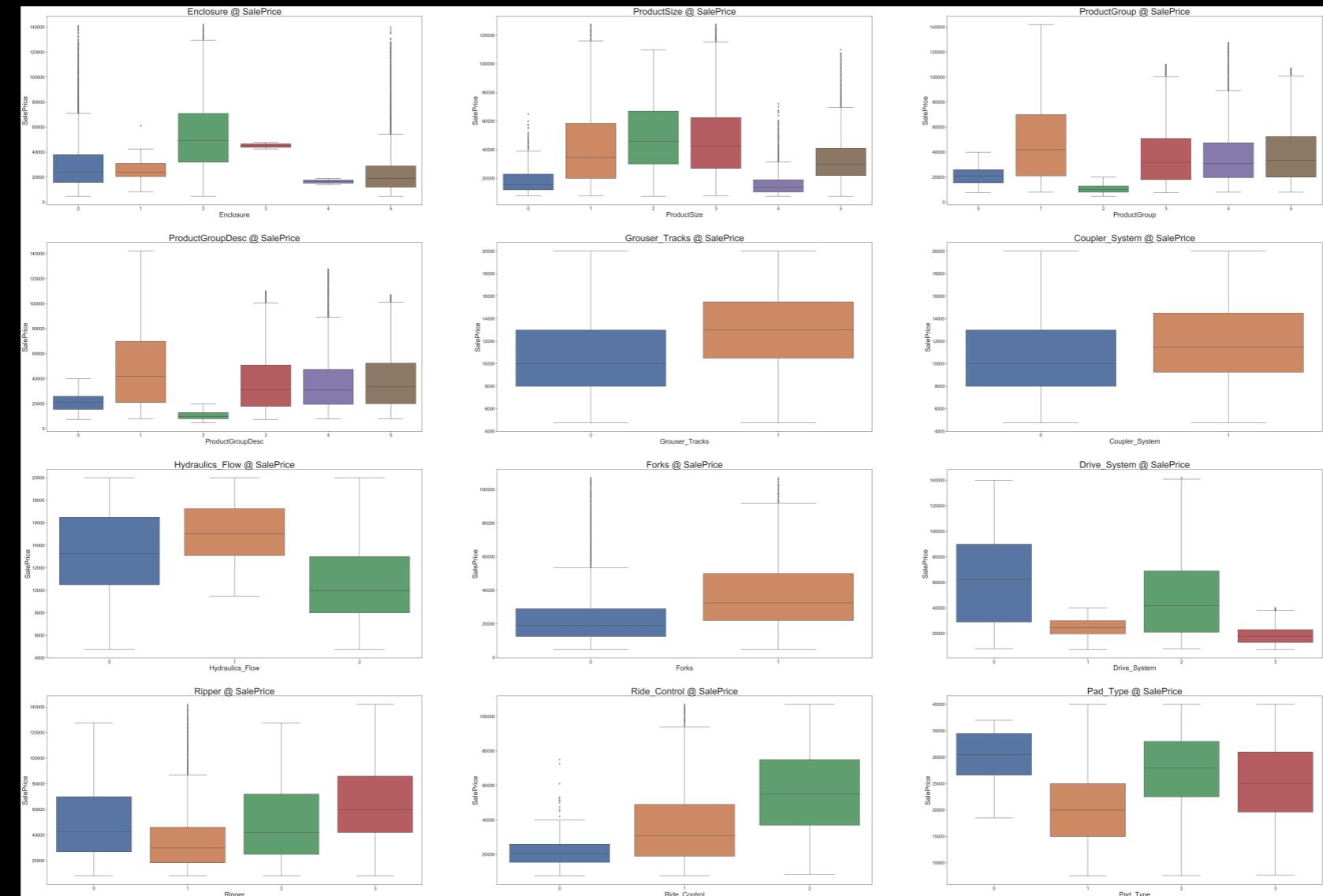
When there is a plethora of variables, it's usually a good call to remove few of those features which are not that relevant to the problem statement at hand. But how to figure those out?. In this project I have used following methodologies to select features, there are plenty more out there...

- Anova η^2 [continuous-discrete] - To gauge how associated a categorical feature is on Target Variable.



SALE PREDICTION

Feature Selection



BoxPlots

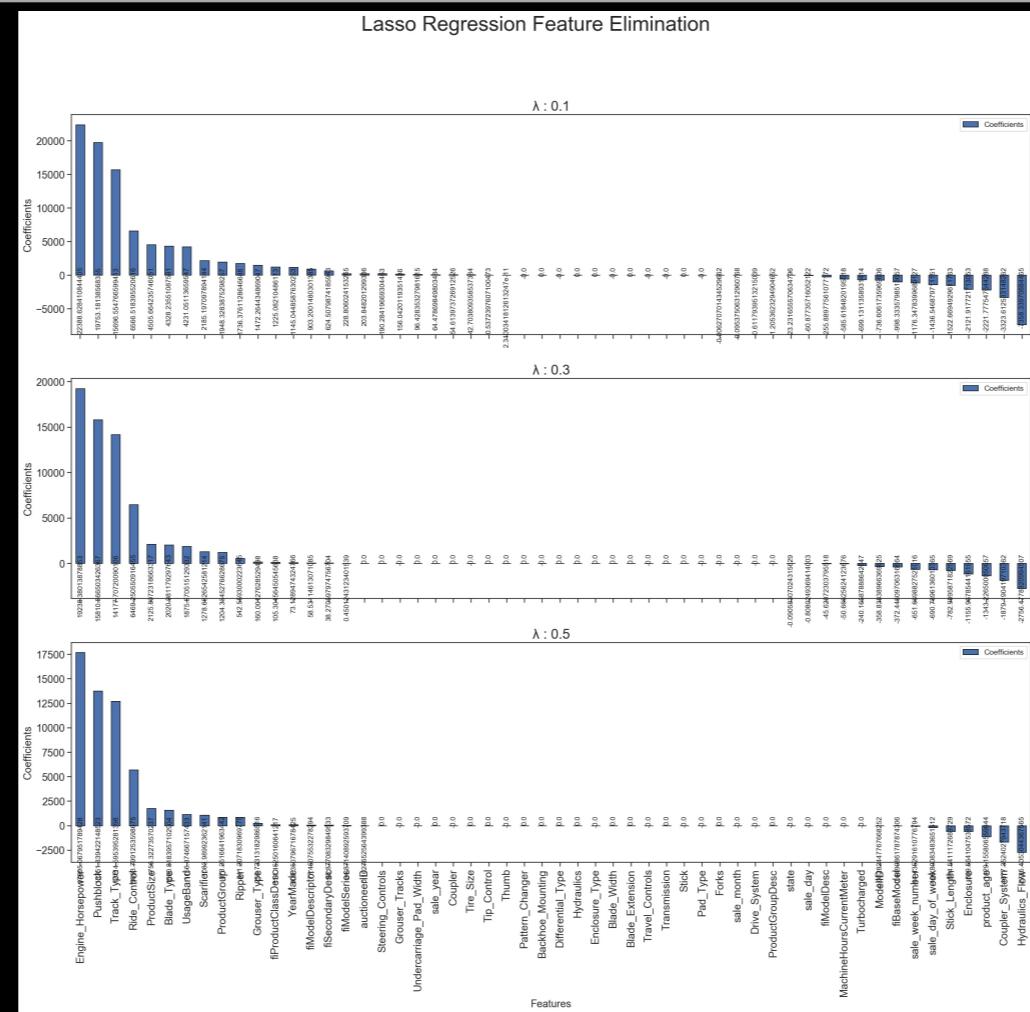
When there is a plethora of variables, it's usually a good call to remove few of those features which are not that relevant to the problem statement at hand. But how to figure those out?. In this project I have used following methodologies to select features, there are plenty more out there...

- BoxPlots [continuous-discrete] – To filter for those features that have significant variation in their mean across classes wrt to Target variable.



SALE PREDICTION

Feature Selection



**Lasso Regression Weights
varying Lambda**

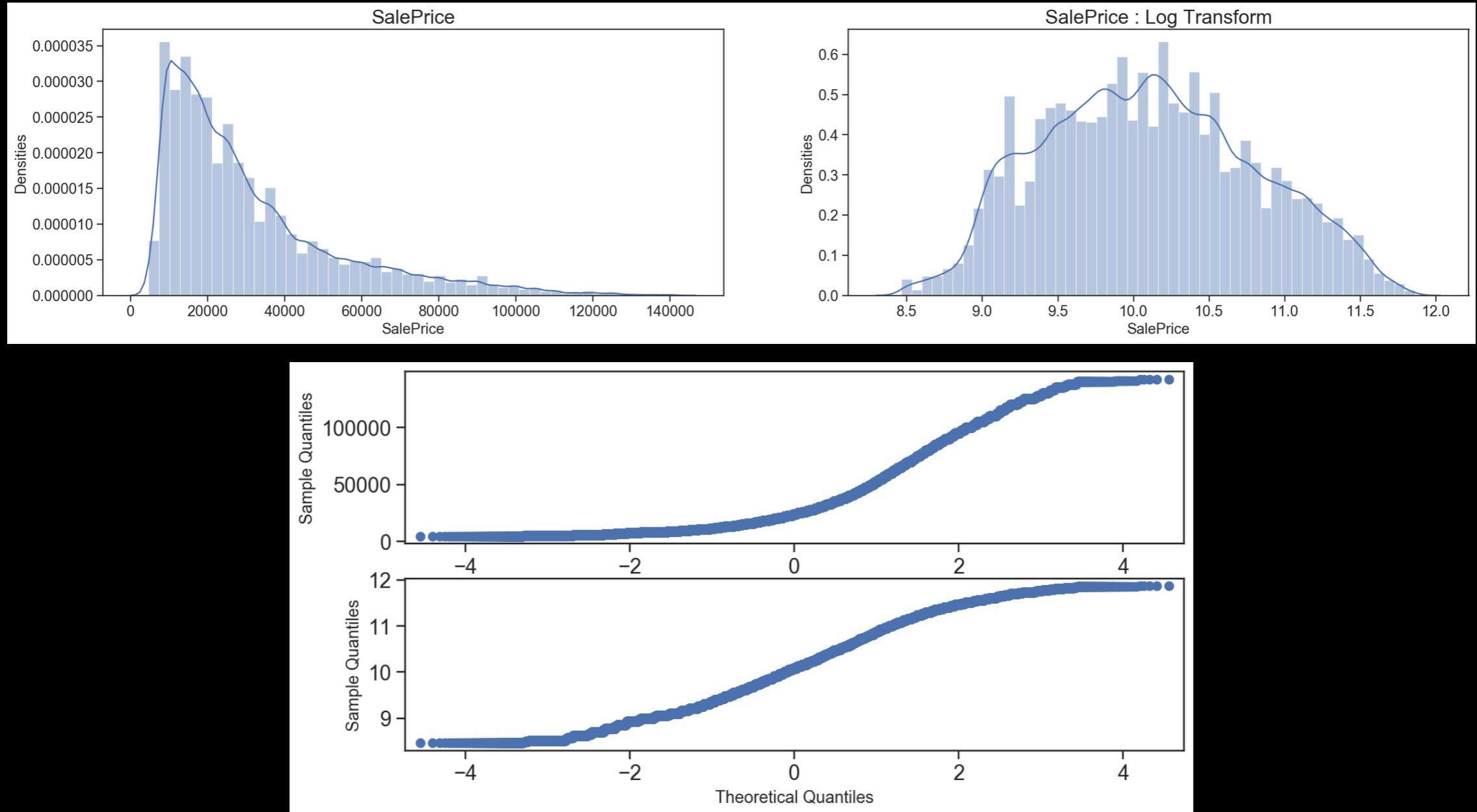
When there is a plethora of variables, it's usually a good call to remove few of those features which are not that relevant to the problem statement at hand. But how to figure those out?. In this project I have used following methodologies to select features, there are plenty more out there...

- Lasso Regression [Feature Elimination] - Eliminating features based on lasso regression.



SALE PREDICTION

EDA



Exploratory Data Analysis is usually takes up substantial effort in the Machine Learning exercise, although in this particular case, I restricted it to just analysing the Target Variable – ‘SalePrice’.

Analysing the Target Variable in the raw form, it was clear that the data is skewed towards the left, applying a simple log transform on the same brought it closer to a Gaussian distribution plot and the Q-Q plot also seem to be *almost* linear to an ideal normal distribution



SALE PREDICTION

Modeling Pipeline

7 Modelling Pipeline

This class contains the entire pipeline for building base models

Data Ingestion -> Data Cleaning/Transformation -> Feature Engineering -> Modelling

7.1 Read Data

```
In [215]: 1 complete_data = pd.read_csv(complete_data_path, index_col=0, low_memory=False)
2 split_ratio = 0.05
3 test_datapoints = int(split_ratio * complete_data.shape[0])
4 TRAIN_DATA = complete_data[:test_datapoints]
5 TEST_DATA = complete_data[test_datapoints:]
6
7 selectedFeatures = pd.read_csv(notebook_results_path+'selectedFeatures.csv', index_col=0).values.ravel().tolist()
8 print('Train Data Shape : ', TRAIN_DATA.shape)
9 print('Test Data Shape : ', TEST_DATA.shape)
```

7.2 Modelling Pipeline

```
In [210]: 1 ▼ class MLPipeline:  
2  
3 ▶     def __init__(self, verbose = False):  
4  
5 ▶     def ingestData(self, trainData, testData, targetVar):  
6  
7 ▶     def cleanTransformData(self, selectedFeatures, fe_dependent_cols):  
8  
9 ▶     def modelling(self, modelDictL1, metrics, splitter, backTransform = True,)  
10  
11 ▶     def evaluateModel(self, metrics, savedModelPath = None):  
12  
13 ▶     def calcMetrics(self, yTrue, yPred, _metrics):  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2079  
2080  
2081  
2082
```

7.3 Base Modelling

[...]

7.4 Fine Tuning - HyperParameters

[...]

7.5 Predicting the Test Set

[...]

The idea behind **Modelling Pipeline** is that we bring in all the requisite functions in one place so that ingesting, cleaning, transforming, base modelling on various models, hyper-parameter of those models, and once done selecting the model evaluate it on the test set.

SALE PREDICTION

Modeling Pipeline

7.3 Base Modelling

```
In [*]: 1 models_L1 = { 'XGBRegressor' : True,
2   'CatBoostRegressor' : True,
3   'LGBMRegressor' : True,
4   'GradientBoostingRegressor' : True,
5   'RandomForestRegressor' : True,
6   'KNeighborsRegressor': True}
7
8 metrics = {'RMSLE':True,
9   'MAPE':True,
10  'MSE':True,
11  'R2':True}
12
13 fe_dependent_cols = ['saledate', 'YearMade']
14
15 kfold_BM = KFold(n_splits=10, shuffle=True, random_state=1)
16
17 MLPipelineHandler = MLPipeline(verbose=True)
18 MLPipelineHandler.ingestData(TRAIN_DATA, TEST_DATA, TARGET_VARIABLE)
19 MLPipelineHandler.cleanTransformData(selectedFeatures, fe_dependent_cols)
20 SCORECARD = MLPipelineHandler.modelling(modeldictL1 = models_L1,
21                                         metrics = metrics,
22                                         splitter = kfold_BM)
23
24
```

execution queued 18:01:03 2019-12-23

Initialising Modelling Pipeline

Initialising Modelling Pipeline...
INGESTION #####
Ingesting Data into Pipeline...
#####

CLEANING #####
Cleaning Data...
Converting dtypes...
Filling missing values with -999...
Engineering Features...
Feature Transformation...
Converting dtypes...
Filling missing values with -999...
Engineering Features...
Feature Transformation...

Data Prepared!
#####

MODELLING

Training XGBRegressor ...
Evaluating for Fold 1
Evaluating for Fold 2
Evaluating for Fold 3
Evaluating for Fold 4
Evaluating for Fold 5
Evaluating for Fold 6
Evaluating for Fold 7
Evaluating for Fold 8
Evaluating for Fold 9
Evaluating for Fold 10

RMSLE : 0.30357745580017526 | MAPE : 0.023257140852327437 | MSE : 0.09216010115005957 | R2 : 0.8070172236422802

BackTransformed-RMSLE : 11015.413827267217 | BackTransformed-MAPE : 664.3560401927249 | BackTransformed-MSE : 121344123.14474602 | BackTransformed-R2 : 0.7689280835225192

Training CatBoostRegressor ...
Evaluating for Fold 1
Evaluating for Fold 2
Evaluating for Fold 3
Evaluating for Fold 4
Evaluating for Fold 5
Evaluating for Fold 6
Evaluating for Fold 7
Evaluating for Fold 8
Evaluating for Fold 9
Evaluating for Fold 10

RMSLE : 0.2456026660905764 | MAPE : 0.01826424298320438 | MSE : 0.060321000253312954 | R2 : 0.8736855146107303

BackTransformed-RMSLE : 8469.170941373633 | BackTransformed-MAPE : 508.6891612599351 | BackTransformed-MSE : 71560425.52436468 | BackTransformed-R2 : 0.8637277459661947

Training LGBMRegressor ...
Evaluating for Fold 1
Evaluating for Fold 2
Evaluating for Fold 3
Evaluating for Fold 4
Evaluating for Fold 5
Evaluating for Fold 6
Evaluating for Fold 7
Evaluating for Fold 8
Evaluating for Fold 9
Evaluating for Fold 10

RMSLE : 0.2493709310316486 | MAPE : 0.01864028174545112 | MSE : 0.062186833279886854 | R2 : 0.8697761664043926

BackTransformed-RMSLE : 8698.39889612255 | BackTransformed-MAPE : 523.8171151220251 | BackTransformed-MSE : 75664809.93296461 | BackTransformed-R2 : 0.8559108112009722

Training GradientBoostingRegressor ...
Evaluating for Fold 1
Evaluating for Fold 2
Evaluating for Fold 3
Evaluating for Fold 4
Evaluating for Fold 5
Evaluating for Fold 6
Evaluating for Fold 7
Evaluating for Fold 8
Evaluating for Fold 9
Evaluating for Fold 10

RMSLE : 0.3031737122688127 | MAPE : 0.023211846582722857 | MSE : 0.09191556470089533 | R2 : 0.807530840804629

BackTransformed-RMSLE : 11003.52580259459 | BackTransformed-MAPE : 663.1908871379671 | BackTransformed-MSE : 121082357.69975547 | BackTransformed-R2 : 0.7694281226422579

Training RandomForestRegressor ...
Evaluating for Fold 1
Evaluating for Fold 2
Evaluating for Fold 3
Evaluating for Fold 4
Evaluating for Fold 5
Evaluating for Fold 6
Evaluating for Fold 7
Evaluating for Fold 8
Evaluating for Fold 9
Evaluating for Fold 10

RMSLE : 0.21703780217080176 | MAPE : 0.01536777080302281 | MSE : 0.04710675376829128 | R2 : 0.9013559287445311

BackTransformed-RMSLE : 7285.101091028926 | BackTransformed-MAPE : 427.0531064475372 | BackTransformed-MSE : 53076618.10543309 | BackTransformed-R2 : 0.8989316934518614

#####



SALE PREDICTION

Modeling Pipeline

8.4 Fine Tuning - HyperParameters

```
In [7]: 1 models_L1 = {'RandomForestRegressor' : True}
2 finetune_Params = {'RandomForestRegressor' : {'n_estimators' : 100, # Number of estimators
3                                                 'criterion' : 'mse',
4                                                 'max_depth' : 5, # Maximum Depth of the trees to build
5                                                 'max_features' : 'auto' # Can take 'auto', 'sqrt', 'log2'
6                                         }
7                                         }
8
9 metrics = {'RMSLE':True,
10          'MAPE':True,
11          'MSE':True,
12          'R2':True}
13
14 fe_dependent_cols = ['saledate', 'YearMade']
15
16 kfold_BM = KFold(n_splits=2, shuffle=True, random_state=1)
17
18 MLPipelineHandler = MLPipeline(verbose=True)
19 MLPipelineHandler.ingestData(TRAIN_DATA,
20                               TEST_DATA,
21                               TARGET_VARIABLE)
22
23 MLPipelineHandler.cleanTransformData(selectedFeatures,
24                                     fe_dependent_cols)
25
26 SCORECARD = MLPipelineHandler.modelling(modelDictL1 = models_L1,
27                                           metrics = metrics,
28                                           splitter = kfold_BM,
29                                           finetune_mode = True,
30                                           finetune_paramdict = finetune_Params,
31                                           finetune_savemodel=True,
32                                           finetunemode_savepath=notebook_results_path)
33
```

executed in 1m 17.7s, finished 19:34:49 2019-12-23

```
Initialising Modelling Pipeline...
#####
##### INGESTION #####
Ingesting Data into Pipeline...
#####

#####
##### CLEANING #####
Cleaning Data...
Converting dtypes...
Filling missing values with -999...
Engineering Features...
Feature Transformation...

Data Prepared!
#####

#####
##### MODELLING #####
Training RandomForestRegressor ...
#####
##### BUILDING FINAL MODEL #####
Loading Model..
Model saved at : ../Notebook Results/RandomForestRegressor-finetunedModel.pkl
#####
```



SALE PREDICTION

Modeling Pipeline

8.5 Predicting the Test Set

```
In [*]: 1 models_L1 = {'RandomForestRegressor' : True}
2
3 metrics = {'RMSLE' : True,
4            'MAPE' : True,
5            'MSE' : True,
6            'R2' : True}
7
8 fe_dependent_cols = ['saledate', 'YearMade']
9
10 MLPipelineHandler = MLPipeline(verbose=True)
11 MLPipelineHandler.ingestData(TRAIN_DATA,
12                             TEST_DATA,
13                             TARGET_VARIABLE)
14
15 MLPipelineHandler.cleanTransformData(selectedFeatures,
16                                       fe_dependent_cols)
17
18 model = MLPipelineHandler.evaluateModel(metrics = metrics,
19                                         savedModelPath = '../Notebook Results/RandomForestRegressor-finetunedMod
20
execution queued 19:35:50 2019-12-23
```

Initialising Modelling Pipeline...

```
##### INGESTION #####
Ingesting Data into Pipeline...
#####
```

```
##### CLEANING #####
Cleaning Data...
Converting dtypes...
Filling missing values with -999...
Engineering Features...
Feature Transformation...
Converting dtypes...
Filling missing values with -999...
Engineering Features...
Feature Transformation...
```

Data Prepared!

```
#####
```

Test Results :

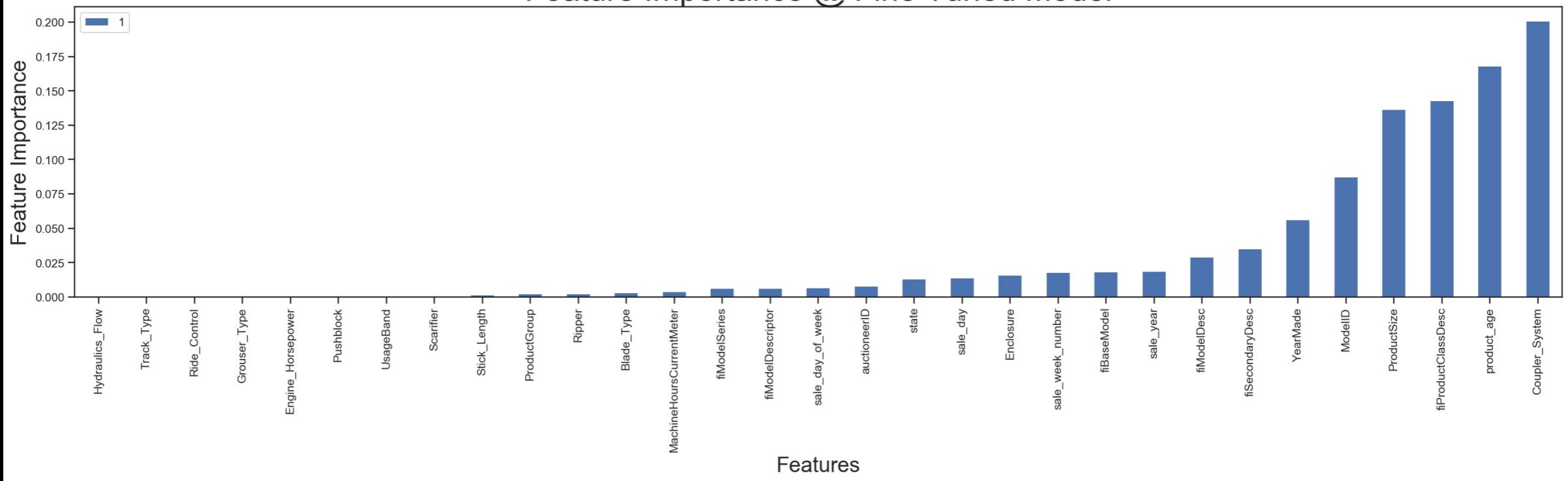
RMSLE : 0.4696427915839575 | MAPE : 0.03623360310026743 | MSE : 0.22056435168677252 | R2 : 0.5969918478672901



SALE PREDICTION

Model Interpretability

Feature Importance @ Fine Tuned Model



Feature Importance

Feature importance for a model



SALE PREDICTION

Model Interpretability

Weight	Feature
0.2010 ± 0.0019	Coupler_System
0.1681 ± 0.0068	product_age
0.1430 ± 0.0177	fiProductClassDesc
0.1364 ± 0.0104	ProductSize
0.0872 ± 0.0082	ModelID
0.0562 ± 0.0064	YearMade
0.0351 ± 0.0032	fiSecondaryDesc
0.0292 ± 0.0097	fiModelDesc
0.0187 ± 0.0010	sale_year
0.0182 ± 0.0095	fiBaseModel
0.0179 ± 0.0004	sale_week_number
0.0158 ± 0.0218	Enclosure
0.0138 ± 0.0004	sale_day
0.0131 ± 0.0004	state
0.0080 ± 0.0003	auctioneerID
0.0068 ± 0.0003	sale_day_of_week
0.0064 ± 0.0020	fiModelDescriptor
0.0063 ± 0.0011	fiModelSeries
0.0039 ± 0.0003	MachineHoursCurrentMeter
0.0032 ± 0.0016	Blade_Type
... 11 more ...	

ELI5 - Global Interpretation

Interpreting the model globally on how important a particular feature is for the model with respective confidence interval.

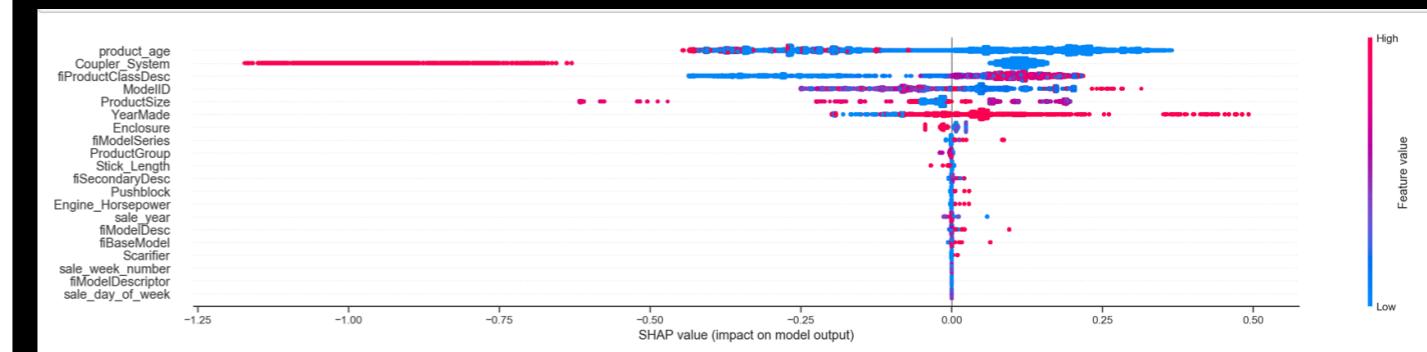
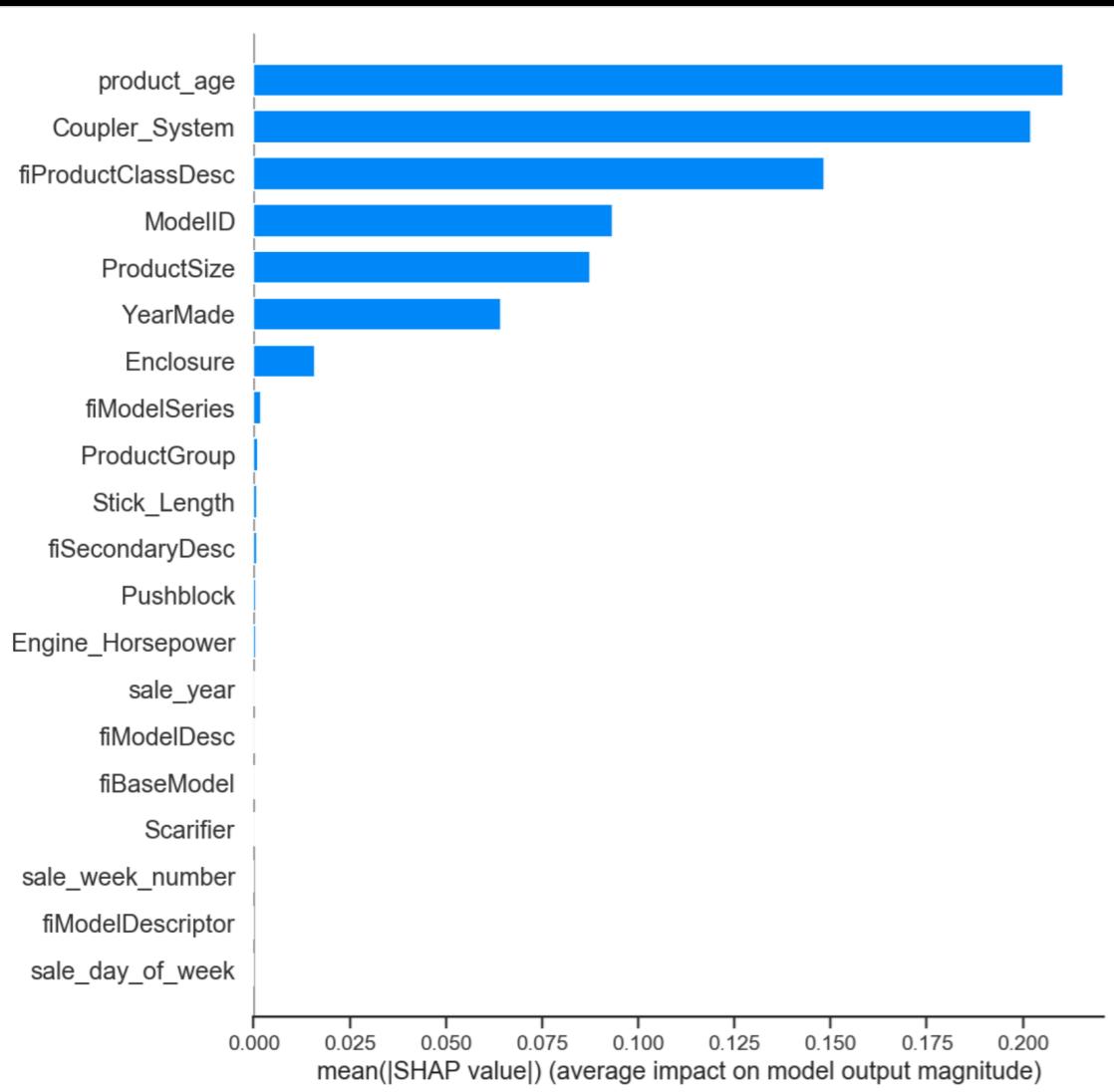
y (score 11.175) top features	
Contribution?	Feature
+10.102	<BIAS>
+0.316	ProductSize
+0.310	product_age
+0.203	fiProductClassDesc
+0.107	Coupler_System
+0.096	sale_week_number
+0.091	fiModelSeries
+0.065	Enclosure
+0.018	fiBaseModel
+0.016	fiSecondaryDesc
+0.015	fiModelDesc
+0.010	sale_day
+0.009	Stick_Length
+0.007	state
+0.006	sale_day_of_week
+0.004	Ripper
+0.003	auctioneerID
+0.002	Grouser_Type
+0.002	fiModelDescriptor
+0.001	ProductGroup
+0.001	Track_Type
+0.000	Blade_Type
+0.000	Ride_Control
-0.001	Scarfier
-0.002	UsageBand
-0.003	Pushblock
-0.003	Engine_Horsepower
-0.008	MachineHoursCurrentMeter
-0.017	YearMade
-0.041	sale_year
-0.133	ModelID

ELI5 - Local Interpretation

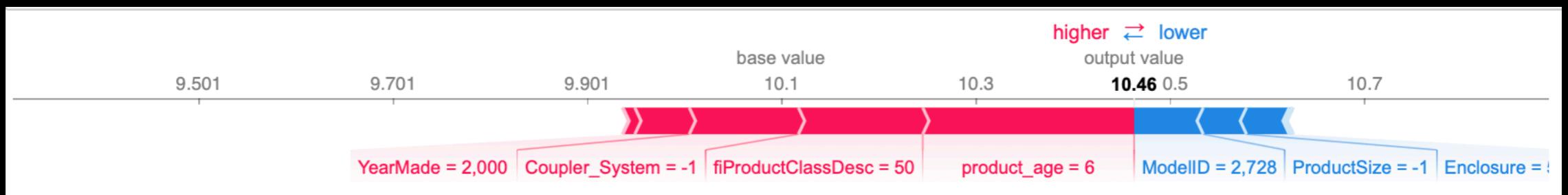
Interpreting the model for individual datapoint a particular feature is for the model.

SALE PREDICTION

Model Interpretability



SHAP - Global



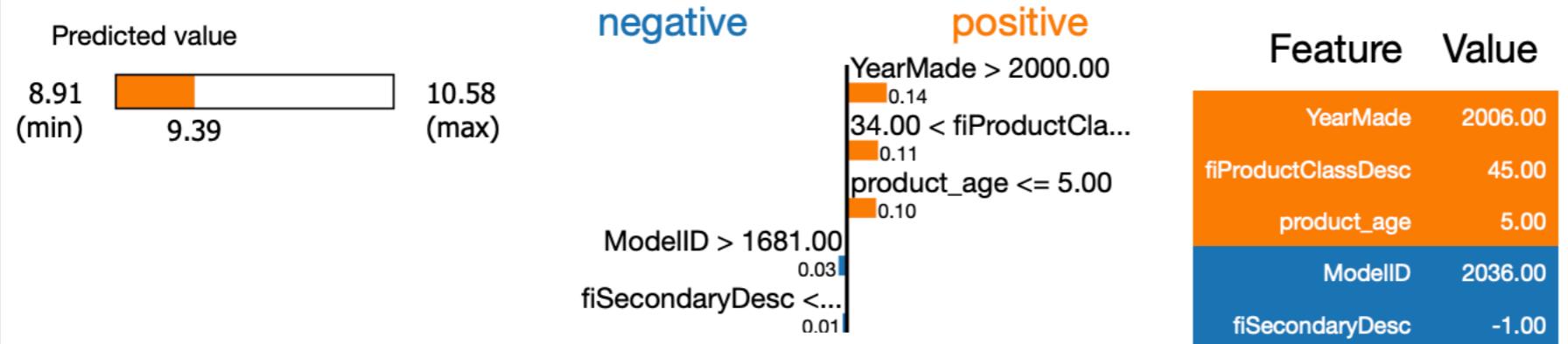
SHAP - Local



SALE PREDICTION

Model Interpretability

```
Intercept 9.045397386616157
Prediction_local [9.35118702]
Right: 9.385470803567575
```



LIME

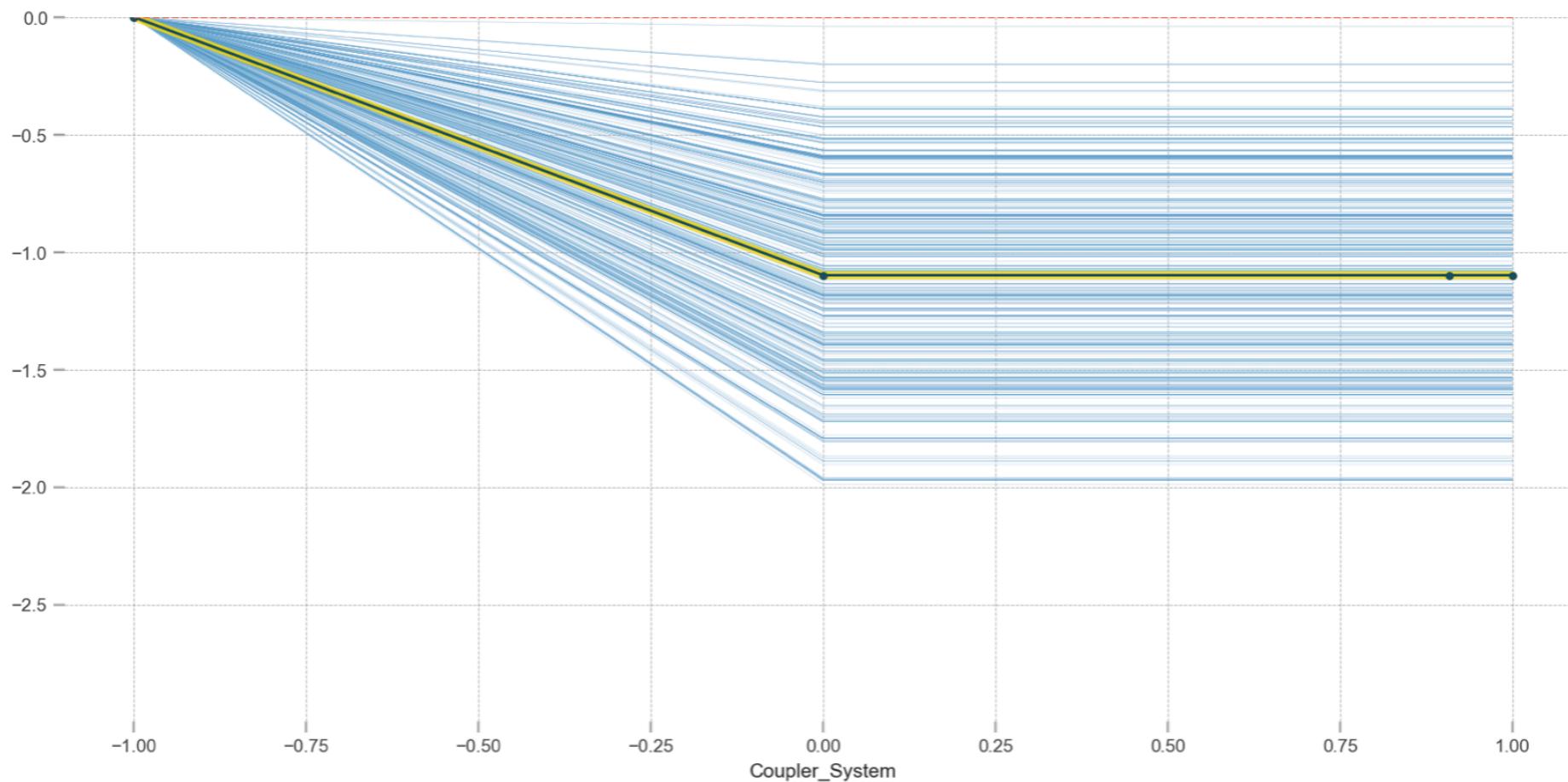


SALE PREDICTION

Model Interpretability

PDP for feature "Coupler_System"

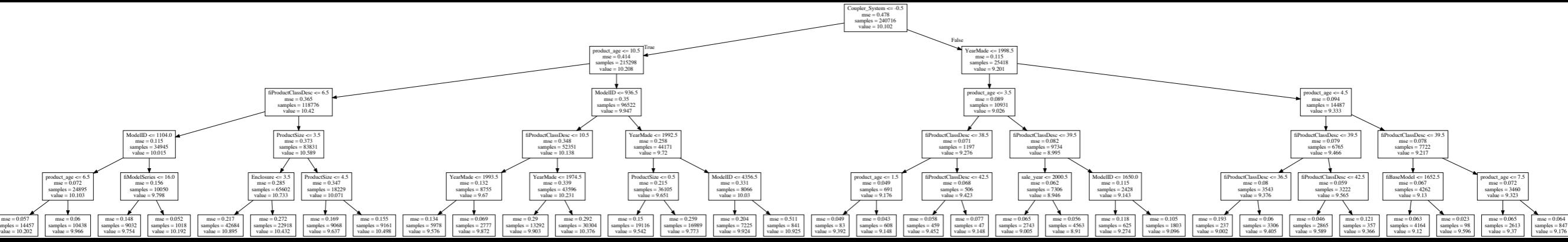
Number of unique grid points: 5



PDP & ICE

SALE PREDICTION

Model Interpretability



Decision Tree Visualisation